Notes for Professionals HTML5 HTML5 Notes for Professionals

100+ pages

of professional hints and tricks

GoalKicker.com Disclaimer

This is an uno?cial free book created for educational purposes and is

Free Programming Books not a?liated with o?cial HTML5 group(s) or company(s).

All trademarks and registered trademarks are the property of their respective owners

Contents

About
……
1

Chapter 1: Getting started with HTML
……
2

Section 1.1: Hello World
……
2

Chapter 2: Doctypes
……
5

Section 2.1: Adding the Doctype
……
5

Section 2.2: HTML 5 Doctype
……
5

Chapter 3: Headings
……
6

Section 3.1: Using Headings
.....
6

Chapter 4: Paragraphs
.....
7

Section 4.1: HTML Paragraphs
.....
7

Chapter 5: Text Formatting
.....
8

Section 5.1: Highlighting
.....
8

Section 5.2: Bold, Italic, and Underline
.....
8

Section 5.3: Abbreviation
.....
9

Section 5.4: Inserted, Deleted, or Stricken
.....
9

Section 5.5: Superscript and Subscript
.....
9

Chapter 6: Anchors and Hyperlinks
.....
11

Section 6.1: Link to another site
.....
11

Section 6.2: Link to an anchor
.....

12

Section 6.3: Link to a page on the same site
.....
12

Section 6.4: Link that dials a number
.....
12

Section 6.5: Open link in new tab/window
.....
13

Section 6.6: Link that runs JavaScript
.....
13

Section 6.7: Link that runs email client
.....
14

Chapter 7: Lists
.....
15

Section 7.1: Ordered List
.....
15

Section 7.2: Unordered List
.....
16

Section 7.3: Nested lists
.....
17

Section 7.4: Description List
.....
17

Chapter 8: Tables
.....
19

Section 8.1: Simple Table 19

Section 8.2: Spanning columns or rows 19

Section 8.3: Column Groups 20

Section 8.4: Table with thead, tbody, tfoot, and caption 21

Section 8.5: Heading scope 22

Chapter 9: Comments 24

Section 9.1: Creating comments 24

Section 9.2: Commenting out whitespace between inline elements 24

Chapter 10: Classes and IDs 26

Section 10.1: Giving an element a class 26

Section 10.2: Giving an element an ID 27

Section 10.3: Acceptable Values 27

Section 10.4: Problems related to duplicated IDs 29

Chapter 11: Data Attributes 30

Section 11.1: Older browsers support 30

Section 11.2: Data Attribute Use 30

Chapter 12: Linking Resources 31

Section 12.1: JavaScript 31

Section 12.2: External CSS Stylesheet 32

Section 12.3: Favicon 32

Section 12.4: Alternative CSS 32

Section 12.5: Resource Hint: dns-prefetch, prefetch, prerender 33

Section 12.6: Link ‘media’ attribute 33

Section 12.7: Prev and Next 33

Section 12.8: Web Feed
……
33

Chapter 13: Include JavaScript Code in HTML
……
35

Section 13.1: Handling disabled Javascript
……
35

Section 13.2: Linking to an external JavaScript file
……..
35

Section 13.3: Directly including JavaScript code
……
35

Section 13.4: Including a JavaScript file executing asynchronously
…… 35

Chapter 14: Using HTML with CSS
……
36

Section 14.1: External Stylesheet Use
……
36

Section 14.2: Internal Stylesheet
……
36

Section 14.3: Inline Style
……
37

Section 14.4: Multiple Stylesheets
……
37

Chapter 15: Images
……
38

Section 15.1: Creating an image 38

Section 15.2: Choosing alt text 38

Section 15.3: Responsive image using the srcset attribute 39

Section 15.4: Responsive image using picture element 40

Chapter 16: Image Maps 41

Section 16.1: Introduction to Image Maps 41

Chapter 17: Input Control Elements 43

Section 17.1: Text 43

Section 17.2: Checkbox and Radio Buttons 44

Section 17.3: Input Validation 46

Section 17.4: Color 47

Section 17.5: Password 48

Section 17.6: File
……
48

Section 17.7: Button
……
49

Section 17.8: Submit
……
50

Section 17.9: Reset
……
50

Section 17.10: Hidden
……
50

Section 17.11: Tel
……
50

Section 17.12: Email
……
51

Section 17.13: Number
……
51

Section 17.14: Range
……
51

Section 17.15: Search
……
51

Section 17.16: Image
……
51

Section 17.17: Week
……

52

Section 17.18: Url
52

Section 17.19: DateTime-Local
52

Section 17.20: Month
52

Section 17.21: Time
52

Section 17.22: DateTime (Global)
53

Section 17.23: Date
53

Chapter 18: Forms
54

Section 18.1: Submitting
54

Section 18.2: Target attribute in form tag
55

Section 18.3: Uploading Files
55

Section 18.4: Grouping a few input fields
55

Chapter 19: Div Element
.....
57

Section 19.1: Basic usage
.....
57

Section 19.2: Nesting
.....
57

Chapter 20: Sectioning Elements
.....
59

Section 20.1: Nav Element
.....
59

Section 20.2: Article Element
.....
60

Section 20.3: Main Element
.....
61

Section 20.4: Header Element
.....
62

Section 20.5: Footer Element
.....
63

Section 20.6: Section Element
.....
63

Chapter 21: Navigation Bars
.....
64

Section 21.1: Basic Navigation Bar
.....

64

Section 21.2: HTML5 Navigation Bar
……
64

Chapter 22: Label Element
……
65

Section 22.1: About Label
……
65

Section 22.2: Basic Use
……
65

Chapter 23: Output Element
……
67

Section 23.1: Output Element Using For and Form Attributes
…….. 67

Section 23.2: Output Element with Attributes
……
67

Chapter 24: Void Elements
……
68

Section 24.1: Void elements
……
68

Chapter 25: Media Elements
……
69

Section 25.1: Audio
……
69

Section 25.2: Video

…… 69

Section 25.3: Using `<video>` and `<audio>` element to display audio/video content …….. 69

Section 25.4: Video header or background …… 70

Chapter 26: Progress Element …… 71

Section 26.1: Progress …… 71

Section 26.2: Changing the color of a progress bar …… 71

Section 26.3: HTML Fallback …… 72

Chapter 27: Selection Menu Controls …… 73

Section 27.1: Select Menu …… 73

Section 27.2: Options …… 73

Section 27.3: Option Groups …… 74

Section 27.4: Datalist …… 74

Chapter 28: Embed
.....
76

Section 28.1: Basic usage
.....
76

Section 28.2: Defining the MIME type
.....
76

Chapter 29: IFrames
.....
77

Section 29.1: Basics of an Inline Frame
.....
77

Section 29.2: Sandboxing
.....
77

Section 29.3: Setting the Frame Size
.....
77

Section 29.4: Using the “srcdoc” Attribute
.....
78

Section 29.5: Using Anchors with IFrames
.....
78

Chapter 30: Content Languages
.....
79

Section 30.1: Base Document Language
.....
79

Section 30.2: Element Language
.....

79

Section 30.3: Elements with Multiple Languages
……..
79

Section 30.4: Regional URLs
……
79

Section 30.5: Handling Attributes with Di?erent Languages
……. 79

Chapter 31: SVG
……
81

Section 31.1: Inline SVG
……
81

Section 31.2: Embedding external SVG files in HTML
……..
81

Section 31.3: Embedding SVG using CSS
……
82

Chapter 32: Canvas
……
83

Section 32.1: Basic Example
……
83

Section 32.2: Drawing two rectangles on a <canvas>
……
83

Chapter 33: Meta Information
……
85

Section 33.1: Page Information

85

Section 33.2: Character Encoding
85

Section 33.3: Robots
86

Section 33.4: Social Media
86

Section 33.5: Mobile Layout Control
87

Section 33.6: Automatic Refresh
88

Section 33.7: Phone Number Recognition
88

Section 33.8: Automatic redirect
88

Section 33.9: Web App
89

Chapter 34: Marking up computer code
90

Section 34.1: Block with <pre> and <code>
90

Section 34.2: Inline with <code>
90

Chapter 35: Marking-up Quotes 91

Section 35.1: Inline with <q> 91

Section 35.2: Block with <blockquote> 91

Chapter 36: Tabindex 93

Section 36.1: Add an element to the tabbing order 93

Section 36.2: Remove an element from the tabbing order 93

Section 36.3: Define a custom tabbing order (not recommended) 93

Chapter 37: Global Attributes 94

Section 37.1: Contenteditable Attribute 94

Chapter 38: HTML 5 Cache 95

Section 38.1: Basic Example of HTML5 cache 95

Chapter 39: HTML Event Attributes 96

Section 39.1: HTML Form Events
……
96

Section 39.2: Keyboard Events
……
96

Chapter 40: Character Entities
……
97

Section 40.1: Character Entities in HTML
……
97

Section 40.2: Common Special Characters
……
97

Chapter 41: ARIA
……
98

Section 41.1: role = “presentation”
……
98

Section 41.2: role = “alert”
……
98

Section 41.3: role = “alertdialog”
……
98

Section 41.4: role = “application”
……
98

Section 41.5: role = “article”
……
98

Section 41.6: role = “banner”
……

99

Section 41.7: role = “button”
.....
99

Section 41.8: role = “cell”
.....
99

Section 41.9: role = “checkbox”
.....
99

Section 41.10: role = “columnheader”
.....
100

Section 41.11: role = “combobox”
.....
100

Section 41.12: role = “complementary”
.....
100

Section 41.13: role = “contentinfo”
.....
100

Section 41.14: role = “definition”
.....
100

Section 41.15: role = “dialog”
.....
101

Section 41.16: role = “directory”
.....
101

Section 41.17: role = “document”
.....
101

Section 41.18: role = “form” 101

Section 41.19: role = “grid” 102

Section 41.20: role = “gridcell” 102

Section 41.21: role = “group” 102

Section 41.22: role = “heading” 102

Section 41.23: role = “img” 103

Section 41.24: role = “link” 103

Section 41.25: role = “list” 103

Section 41.26: role = “listbox” 103

Section 41.27: role = “listitem” 103

Section 41.28: role = “log” 104

Section 41.29: role = “main” .....

104

Section 41.30: role = “marquee”
104

Section 41.31: role = “math”
104

Section 41.32: role = “menu”
104

Section 41.33: role = “menubar”
104

Section 41.34: role = “menuitem”
105

Section 41.35: role = “menuitemcheckbox”
105

Section 41.36: role = “menuitemradio”
105

Section 41.37: role = “navigation”
105

Section 41.38: role = “note”
105

Section 41.39: role = “option”
105

Section 41.40: role = “progressbar”
106

Section 41.41: role = “radio” 106

Section 41.42: role = “region” 106

Section 41.43: role = “radiogroup” 106

Section 41.44: role = “row” 106

Section 41.45: role = “rowgroup” 107

Section 41.46: role = “rowheader” 107

Section 41.47: role = “scrollbar” 107

Section 41.48: role = “search” 107

Section 41.49: role = “searchbox” 108

Section 41.50: role = “separator” 108

Section 41.51: role = “slider” 108

Section 41.52: role = “spinbutton” .....

108

Section 41.53: role = “status”
……
108

Section 41.54: role = “switch”
……
108

Section 41.55: role = “tab”
……
109

Section 41.56: role = “table”
……
109

Section 41.57: role = “tablist”
……
109

Section 41.58: role = “tabpanel”
……
109

Section 41.59: role = “textbox”
……
109

Section 41.60: role = “timer”
……
110

Section 41.61: role = “toolbar”
……
110

Section 41.62: role = “tooltip”
……
110

Section 41.63: role = “tree”
……
110

Section 41.64: role = “treegrid” 111

Section 41.65: role = “treeitem” 111

Credits 112

You may also like 117

About

Please feel free to share this PDF with anyone for free, latest version of this book can be downloaded from: https://goalkicker.com/HTML5Book

This HTML5 Notes for Professionals book is compiled from Stack Overflow

Documentation, the content is written by the beautiful people at Stack Overflow.

Text content is released under Creative Commons BY-SA, see credits at the end of this book whom contributed to the various chapters. Images may be copyright of their respective owners unless otherwise specified

This is an unofficial free book created for educational purposes and is not affiliated with official HTML5 group(s) or company(s) nor Stack Overflow. All trademarks and registered trademarks are the property of their respective company owners

The information presented in this book is not guaranteed to be correct nor accurate, use at your own risk

Please send feedback and corrections to web@petercv.com

Chapter 1: Getting started with HTML

Version Specification Release Date 1.0 N/A 1994-01-01

2.0 RFC 1866 1995-11-24

3.2 W3C: HTML 3.2 Specification 1997-01-14

4.0 W3C: HTML 4.0 Specification 1998-04-24

4.01 W3C: HTML 4.01 Specification 1999-12-24

5 WHATWG: HTML Living Standard 2014-10-28

5.1 W3C: HTML 5.1 Specification 2016-11-01

Section 1.1: Hello World

Introduction

HTML (Hypertext Markup Language) uses a markup system composed of elements which represent specific content. Markup means that with HTML you declare what is presented to a viewer, not how it is presented. Visual representations are defined by Cascading Style Sheets (CSS) and realized by browsers. Still existing elements that

allow for such, like e.g. font, “are entirely obsolete, and must not be used by authors”[1].

HTML is sometimes called a programming language but it has no logic, so is a markup language. HTML tags provide semantic meaning and machine-readability to the content in the page.

An element usually consists of an opening tag (<element_name>), a closing tag (</element_name>), which contain the element’s name surrounded by angle brackets, and the content in between: <element_name>…content…</element_name>

There are some HTML elements that don’t have a closing tag or any contents. These are called void elements. Void elements include <img>, <meta>, <link> and <input>.

Element names can be thought of as descriptive keywords for the content they contain, such as video, audio, table, footer.

A HTML page may consist of potentially hundreds of elements which are then read by a web browser, interpreted and rendered into human readable or audible content on the screen.

For this document it is important to note the difference between elements and tags:

Elements: video, audio, table, footer

Tags: <video>, <audio>, <table>, <footer>, </html>, </body>

Element insight

Let’s break down a tag…

The <p> tag represents a common paragraph.

Elements commonly have an opening tag and a closing tag. The opening tag contains the element's name in angle

 2 brackets ( ). The closing tag is identical to the opening tag with the addition of a forward slash (/) between the opening bracket and the element's name ( ).

Content can then go between these two tags: This is a simple paragraph. .

Creating a simple page

The following HTML example creates a simple "Hello World" web page.

HTML files can be created using any text editor. The files must be saved with a .html or .htm[2] extension in order to be recognized as HTML files.

Once created, this file can be opened in any web browser.

```
<html lang="en">

<head>

<meta charset="UTF-8">

<title>Hello!</title>

</head>

<body>

<h1>Hello World!</h1>

<p>This is a simple paragraph.</p>

</body>

</html>
```

Simple page break down

These are the tags used in the example:

Tag Meaning

Defines the HTML version used in the document. In this case it is HTML5. > See the doctypes topic for more information.

Opens the page. No markup should come after the closing tag (</ html>). The lang attribute declares <html> the primary language of the page using the ISO language codes (en for English).

See the Content Language topic for more information.

Opens the head section, which does not appear in the main browser window but mainly contains <head> information about the HTML document, called metadata. It can also contain imports from external stylesheets and scripts. The closing tag is </head>.

Gives the browser some metadata about the document. The charset attribute declares the character

<meta> encoding. Modern HTML documents should always use UTF-8, even though it is not a requirement. In

HTML, the <meta> tag does not require a closing tag. See the Meta topic for more information.

<title> The title of the page. Text written between this opening and the closing tag (</title>) will be displayed on the tab of the page or in the title bar of the browser.

<body> Opens the part of the document displayed to users, i.e. all the visible or audible content of a page. No content should be added after the closing tag </body>.

GoalKicker.com • HTML5 Notes for Professionals 3 <h1> A level 1 heading for the page.

See headings for more information.

<p> Represents a common paragraph of text.

1. ? HTML5, 11.2 Non-conforming features

2. ? .htm is inherited from the legacy DOS three character file extension limit.

Chapter 2: Doctypes

Doctypes - short for ‘document type’ - help browsers to understand the version of HTML the document is written in for better interpretability. Doctype declarations are not HTML tags and belong at the very top of a document. This topic explains the structure and declaration of various doctypes in HTML.

Section 2.1: Adding the Doctype

The declaration should always be included at the top of the HTML document, before the <html> tag.

Version ? 5

See HTML 5 Doctype for details on the HTML 5 Doctype.

>

Section 2.2: HTML 5 Doctype

HTML5 is not based on SGML (Standard Generalized Markup Language), and therefore does not require a reference to a DTD (Document Type Definition).

HTML 5 Doctype declaration:

>

Case Insensitivity

Per the W3.org HTML 5 DOCTYPE Spec:

A DOCTYPE must consist of the following components, in this order:

1. A string that is an ASCII case-insensitive match for the string “.

therefore the following DOCTYPEs are also valid:

>

>

>

This SO article discusses the topic extensively: Uppercase or lowercase

doctype?

Chapter 3: Headings

HTML provides not only plain paragraph tags, but six separate header tags to indicate headings of various sizes and thicknesses. Enumerated as heading 1 through heading 6, heading 1 has the largest and thickest text while heading 6 is the smallest and thinnest, down to the paragraph level. This topic details proper usage of these tags.

Section 3.1: Using Headings

Headings can be used to describe the topic they precede and they are defined with the <h1> to <h6> tags. Headings support all the global attributes.

<h1> defines the most important heading.

<h6> defines the least important heading.

Defining a heading:

<h1>Heading 1</h1>

<h2>Heading 2</h2>

<h3>Heading 3</h3>

<h4>Heading 4</h4>

<h5>Heading 5</h5>

<h6>Heading 6</h6>

Correct structure matters

Search engines and other user agents usually index page content based on heading elements, for example to create a table of contents, so using the correct structure for headings is important.

In general, an article should have one h1 element for the main title followed by h2 subtitles going down a layer if necessary. If there are h1 elements on a higher level they shoudn't be used to describe any lower level content.

Example document (extra intendation to illustrate hierarchy):

```
<h1>Main title</h1>

<p>Introduction</p>

<h2>Reasons</h2>

<h3>Reason 1</h3>

<p>Paragraph</p>

<h3>Reason 2</h3>

<p>Paragraph</p>

<h2>In conclusion</h2>

<p>Paragraph</p>
```

Chapter 4: Paragraphs

Column Column

 Defines a paragraph
 Inserts a single line break <pre> Defines pre-formatted text

Paragraphs are the most basic HTML element. This topic explains and demonstrates the usage of the paragraph element in HTML.

Section 4.1: HTML Paragraphs

The HTML element defines a paragraph:

This is a paragraph.

This is another paragraph.

Display-

You cannot be sure how HTML will be displayed.

Large or small screens, and resized windows will create different results.

With HTML, you cannot change the output by adding extra spaces or extra lines in your HTML code.

The browser will remove any extra spaces and extra lines when the page is displayed:

This is another paragraph, extra spaces will be removed by browsers

Chapter 5: Text Formatting

While most HTML tags are used to create elements, HTML also provides in-text formatting tags to apply specific text-related styles to portions of text. This topic includes examples of HTML text formatting such as highlighting, bolding, underlining, subscript, and stricken text.

Section 5.1: Highlighting

The <mark> element is new in HTML5 and is used to mark or highlight text in a document “due to its relevance in another context”.1

The most common example would be in the results of a search were the user has entered a search query and results are shown highlighting the desired query.

```
<p>Here is some content from an article that contains the <mark>searched query</mark> that we are looking for. Highlighting the text will make it easier for the user to find what they are looking for.</p>
```

Output:

A common standard formatting is black text on a yellow background, but this can be changed with CSS.

Section 5.2: Bold, Italic, and Underline

Bold Text

To bold text, use the or <b> tags:

```
<strong>Bold Text Here</strong>
```

or

```
<b>Bold Text Here</b>
```

What’s the difference? Semantics. is used to indicate that the text is fundamentally or semantically important to the surrounding text, while <b> indicates no such importance and simply represents text that should be bolded.

If you were to use <b> a text-to-speech program would not say the

word(s) any differently than any of the other words around it - you are simply drawing attention to them without adding any additional importance. By using , though, the same program would want to speak those word(s) with a different tone of voice to convey that the text is important in some way.

Italic Text

To italicize text, use the or tags:

GoalKicker.com HTML5 Notes for Professionals 8 Italicized Text Here

or

Italicized Text Here

What's the difference? Semantics. is used to indicate that the text should have extra emphasis that should be stressed, while simply represents text which should be set off from the normal text around it.

For example, if you wanted to stress the action inside a sentence, one might do so by emphasizing it in italics via : "Would you just submit the edit already?"

But if you were identifying a book or newspaper that you would normally italicize stylistically, you would simply use : "I was forced to read Romeo and Juliet in high school.

Underlined Text

While the <u> element itself was deprecated in HTMl 4, it was reintroduced with alternate semantic meaning in HTML 5 - to represent an unarticulated, non-textual annotation. You might use such a rendering to indicate misspelled text on the page, or for a Chinese proper name mark.

This paragraph contains some <u>mispelled</u> text.

Section 5.3: Abbreviation

To mark some expression as an abbreviation, use <abbr> tag:

I like to write <abbr title="Hypertext Markup Language">HTML</abbr>!

If present, the title attribute is used to present the full description of such abbreviation.

Section 5.4: Inserted, Deleted, or Stricken

To mark text as inserted, use the <ins> tag:

<ins>New Text</ins>

To mark text as deleted, use the <del> tag:

~~Deleted Text~~

To strike through text, use the <s> tag:

<s>Struck-through text here</s>

Section 5.5: Superscript and Subscript

To offset text either upward or downward you can use the tags <sup> and <sub>.

To create superscript:

^{superscript here}

GoalKicker.com • HTML5 Notes for Professionals 9 To create subscript:

_{subscript here}

Chapter 6: Anchors and Hyperlinks

Parameter Details Specifies the destination address. It can be an absolute or relative URL, or the name of an anchor. An absolute URL is the complete URL of a website like http://example.com/. A relative URL points to

href another directory and/or document inside the same website, e.g. /about-us/ points to the directory "about-us" inside the root directory (/). When pointing to another directory without explicitly specifying the document, web servers typically return the document "index.html" inside that directory.

hreflang Specifies the language of the resource linked by the href attribute (which must be present with this one). Use language values from BCP 47 for HTML5 and RFC 1766 for HTML 4.

rel Specifies the relationship between the current document and the linked document. For HTML5, the values must be defined in the specification or registered in the Microformats wiki.

Specifies where to open the link, e.g. in a new tab or window. Possible values are _blank, _self, target _parent, _top, and framename (deprecated). Forcing such behaviour is not recommended since it violates the control of the user over a website.

Specifies extra information about a link. The information is most often shown as a tooltip text when title the cursor moves over the link. This attribute is not restricted to links, it can be used on almost all HTML tags.

Specifies that the target will be downloaded when a user clicks on the hyperlink. The value of the download attribute will be the name of the downloaded file. There are no restrictions on allowed values, and the browser will automatically detect the correct file extension and add it to the file (.img, .pdf, etc.). If the value is omitted, the original filename is used.

Anchor tags are commonly used to link separate webpages, but they can also be used to link between different places in a single document, often within table of contents or even launch external applications. This topic explains the implementation and application of HTML anchor tags in various roles.

Section 6.1: Link to another site

This is the basic use of the <a> (anchor element) element:

Link to example.com

It creates a hyperlink, to the URL http://example.com/ as specified by the href (hypertext reference) attribute, with the anchor text "Link to example.com". It would look something like the following:

Link to example.com

To denote that this link leads to an external website, you can use the external link type:

example site

You can link to a site that uses a protocol other than HTTP. For example, to link to an FTP site, you can do,

This could be a link to a FTP site

In this case, the difference is that this anchor tag is requesting that the user's browser connect to example.com using the File Transfer Protocol (FTP) rather than the Hypertext Transfer Protocol (HTTP).

This could be a link to a FTP site

GoalKicker.com • HTML5 Notes for Professionals 11 Section 6.2: Link to an anchor

Anchors can be used to jump to specific tags on an HTML page. The <a> tag can point to any element that has an id attribute. To learn more about IDs, visit the documentation about Classes and IDs. Anchors are mostly used to jump to a subsection of a page and are used in conjunction with header tags.

Suppose you've created a page (page1.html) on many topics:

<h2>First topic</h2>

Content about the first topic

<h2>Second topic</h2>

Content about the second topic

Once you have several sections, you may want to create a Table of Contents at the top of the page with quick-links (or bookmarks) to specific sections.

If you gave an id attribute to your topics, you could then link to them

<h2 id="Topic1">First topic</h2>

Content about the first topic

<h2 id="Topic2">Second topic</h2>

Content about the second topic

Now you can use the anchor in your table of contents:

<h1>Table of Contents</h1>

Click to jump to the First Topic Click to jump to the Second Topic

These anchors are also attached to the web page they're on (page1.html). So you can link across the site from one page to the other by referencing the page and anchor name.

Remember, you can always look back in the First Topic for supporting information.

Section 6.3: Link to a page on the same site

You can use a relative path to link to pages on the same website.

Text Here

The above example would go to the file example at the root directory (/) of the server.

If this link was on http://example.com, the following two links would bring the user to the same location

Text Here

Text Here

Both of the above would go to the page file at the root directory of example.com.

Section 6.4: Link that dials a number

If the value of the href-attribute begins with tel:, your device will dial the number when you click it. This works on mobile devices or on computers/tablets running software like Skype or FaceTime that can make phone calls.

GoalKicker.com HTML5 Notes for Professionals 12 Call us

Most devices and programs will prompt the user in some way to confirm the number they are about to dial.

Section 6.5: Open link in new tab/window

Text Here

The target attribute specifies where to open the link. By setting it to _blank, you tell the browser to open it in a new tab or window (per user preference).

SECURITY VULNERABILITY WARNING!

Using target="_blank" gives the opening site partial access to the window.opener object via JavaScript, which allows that page to then access and change the window.opener.location of your page and potentially redirect users to malware or phishing sites.

Whenever using this for pages you do not control, add rel="noopener" to your link to prevent the window.opener object from being sent with the request.

Currently, Firefox does not support noopener, so you will need to use rel="noopener noreferrer" for maximum effect.

Section 6.6: Link that runs JavaScript

Simply use the javascript: protocol to run the text as JavaScript instead of opening it as a normal link:

Run Code

You can also achieve the same thing using the onclick attribute:

Run Code

The return false; is necessary to prevent your page from scrolling to the top when the link to # is clicked. Make sure to include all code you'd like to run before it, as returning will stop execution of further code.

Also noteworthy, you can include an exclamation mark ! after the hashtag in order to prevent the page from scrolling to the top. This works because any invalid slug will cause the link to not scroll anywhere on the page, because it couldn't locate the element it references (an element with id = "!"). You could also just use any invalid slug (such as #scrollsNowhere) to achieve the same effect. In this case, return false; is not required:

< a href = "#!" onclick = "myFunction();" > Run Code < /a >

Should you be using any of this?

The answer is almost certainly no. Running JavaScript inline with the element like this is fairly bad practice. Consider using pure JavaScript solutions that look for the element in the page and bind a function to it instead. Listening for an event

Also consider whether this element is really a button instead of a link. If so, you should use < button > .

GoalKicker.com • HTML5 Notes for Professionals 13 Section 6.7: Link that runs email client

Basic usage

If the value of the href-attribute begins with mailto: it will try to open an email client on click:

< a href = "mailto:example@example.com" > Send email < /a >

This will put the email address example@example.com as the recipient for the newly created email.

Cc and Bcc

You can also add addresses for cc- or bcc-recipients using the following syntax:

< a href = "mailto:example@example.com?
cc = john@example.com&bcc = jane@example.com" > Send email < /
a >

Subject and body text

You can populate the subject and body for the new email as well:

```
<a href="mailto:example@example.com?subject=Example
+subject&body=Message+text">Send email</a>
```

Those values must be URL encoded.

Clicking on a link with mailto: will try to open the default email client specified by your operating system or it will ask you to choose what client you want to use. Not all options specified after the recipient's address are supported in all email clients.

Chapter 7: Lists

HTML offers three ways for specifying lists: ordered lists, unordered lists, and description lists. Ordered lists use ordinal sequences to indicate the order of list elements, unordered lists use a defined symbol such as a bullet to list elements in no designated order, and description lists use indents to list elements with their children. This topic explains the implementation and combination of these lists in HTML markup.

Section 7.1: Ordered List

An ordered list can be created with the <ol> tag and each list item can be created with the <li> tag as in the example below:

Item

Another Item

Yet Another Item

This will produce a numbered list (which is the default style):

1. Item

2. Another Item

3. Yet Another Item

Manually changing the numbers

There are a couple of ways you can play with which numbers appear on the list items in an ordered list. The first way is to set a starting number, using the start attribute. The list will start at this defined number, and continue incrementing by one as usual.

<ol start="3">

Item

Some Other Item

Yet Another Item

This will produce a numbered list (which is the default style):

3. Item

4. Some Other Item

5. Yet Another Item

You can also explicitly set a certain list item to a specific number. Further list items after one with a specified value will continue incrementing by one from that list item's value, ignoring where the parent list was at.

<li value="7">

It is also worth noting that, by using the value attribute directly on a list item, you can override an ordered list's existing numbering system by restarting the numbering at a lower value. So if the parent list was already up to value 7, and encountered a list item at value 4, then that list item would still display as 4 and continue counting

 from that point again.

<ol start="5">

Item

Some Other Item

<li value="4">A Reset Item

Another Item

Yet Another Item

So the example above will produce a list that follows the numbering pattern of 5, 6, 4, 5, 6 - starting again at a number lower than the previous and duplicating the number 6 in the list.

Note: The start and value attributes only accept a number - even if the

ordered list is set to display as Roman numerals or letters.

Version ? 5

You can reverse the numbering by adding reversed in your ol element:

<ol reversed>

Item

Some Other Item

<li value="4">A Reset Item

Another Item

Yet Another Item

Reverse numbering is helpful if you're continually adding to a list, such as with new podcast episodes or presentations, and you want the most recent items to appear first.

Changing the type of numeral

You can easily change the type of numeral shown in the list item marker by using the type attribute

<ol type="1|a|A|i|I">

Type Description Examples

1 Default value - Decimal numbers 1,2,3,4

a Alphabetically ordered (lowercase) a,b,c,d A Alphabetically ordered (uppercase) A,B,C,D

i Roman Numerals (lowercase) i,ii,iii,iv I Roman Numerals (uppercase) I,II,III,IV

You should use ol to display a list of items, where the items have been intentionally ordered and order should be emphasized. If changing the order of the items does NOT make the list incorrect, you should use <ul>.

Section 7.2: Unordered List

An unordered list can be created with the <ul> tag and each list item can be created with the <li> tag as shown by the example below:

GoalKicker.com – HTML5 Notes for Professionals 16

Item

Another Item

Yet Another Item

This will produce a bulleted list (which is the default style):

Item

Another Item

Yet Another Item

You should use ul to display a list of items, where the order of the items is not important. If changing the order of the items makes the list incorrect, you should use <ol>.

Section 7.3: Nested lists

You can nest lists to represent sub-items of a list item.

item 1

item 2

sub-item 2.1

sub-item 2.2

item 3

item 1

item 2

sub-item 2.1

sub-item 2.2

item 3

The nested list has to be a child of the li element.

You can nest different types of list, too:

Hello, list!

Hello, nested list!

Section 7.4: Description List

A description list (or definition list, as it was called before HTML5) can be created with the dl element. It consists of

GoalKicker.com • HTML5 Notes for Professionals 17 name-value groups, where the name is given in the dt element, and the value is given in the dd element.

<dl>

<dt>name 1</dt>

<dd>value for 1</dd>

<dt>name 2</dt>

<dd>value for 2</dd>

</dl>

Live demo

A name-value group can have more than one name and/or more than one value (which represent alternatives):

<dl>

<dt>name 1</dt>

<dt>name 2</dt>

<dd>value for 1 and 2</dd>

<dt>name 3</dt>

<dd>value for 3</dd>

<dd>value for 3</dd>

</dl>

Live demo

Chapter 8: Tables

The HTML <table> element allows web authors to display tabular data (such as text, images, links, other tables, etc.) in a two dimensional table with rows and columns of cells.

Section 8.1: Simple Table

<table>

<tr>

<th>Heading 1/Column 1</th>

<th>Heading 2/Column 2</th>

</tr>

<tr>

<td>Row 1 Data Column 1</td>

<td>Row 1 Data Column 2</td>

</tr>

<tr>

<td>Row 2 Data Column 1</td>

<td>Row 2 Data Column 2</td>

</tr>

</table>

This will render a <table> consisting of three total rows (<tr>): one row of header cells (<th>) and two rows of content cells (<td>). <th> elements are tabular headers and <td> elements are tabular data. You can put whatever you want inside a <td> or <th>.

Heading 1/Column 1 Heading 2/Column 2

Row 1 Data Column 1 Row 1 Data Column 2

Row 2 Data Column 1 Row 2 Data Column 2

Section 8.2: Spanning columns or rows

Table cells can span multiple columns or rows using the colspan and rowspan attributes. These attributes can be applied to <th> and <td> elements.

<table>

<tr>

<td>row 1 col 1</td>

<td>row 1 col 2</td>

<td>row 1 col 3</td>

</tr>

<tr>

<td colspan="3">This second row spans all three columns</td>

</tr>

<tr>

<td rowspan="2">This cell spans two rows</td> <td>row 3 col 2</td>

<td>row 3 col 3</td>

</tr>

<tr>

<td>row 4 col 2</td>

<td>row 4 col 3</td>

</tr>

</table>

 Will result in

Note that you should not design a table where both rows and columns overlap as this is invalid HTML and the result is handled differently by different web browsers.

rowspan = A non-negative integer that specifies the number of rows spanned by a cell. The default value of this attribute is one (1). A value of zero (0) means that the cell will extend from the current row until the last row of the table (<thead>, <tbody>, or <tfoot>).

colspan = A non-negative integer that specifies the number of columns spanned by the current cell. The default value of this attribute is one (1). A value of zero (0) means that the cell will extend from the current to the last column of the column group <colgroup> in which the cell is defined.

Section 8.3: Column Groups

Sometimes you may want to apply styling to a column or group of columns. Or for semantic purposes, you may want to group columns together. To do this, use <colgroup> and <col> elements.

The optional <colgroup> tag allows you to group columns together. <colgroup> elements must be child elements of a <table> and must come after any <caption> elements and before any table content (e.g., <tr>, <thead>, <tbody> , etc.).

<table>

<colgroup span="2"></colgroup>

<colgroup span="2"></colgroup>

…

</table>

The optional <col> tag allows you to reference individual columns or a range of columns without applying a logical grouping. <col> elements are optional, but if present, they must be inside a <colgroup> element.

<table>

<colgroup>

<col id="MySpecialColumn" />

<col />

</colgroup>

<colgroup>

<col class="CoolColumn" />

<col class="NeatColumn" span="2" />

</colgroup>

…

</table>

The following CSS styles can be applied to <colgroup> and <col> elements:

border

background

width

visibility

display (as in display: none)

display: none; will actually remove the columns from the display, causing the table to render as if those cells don't exist

For more information, see HTML5 Tabular data.

Section 8.4: Table with thead, tbody, tfoot, and caption

HTML also provides the tables with the <thead>, <tbody>, <tfoot>, and <caption> elements. These additional elements are useful for adding semantic value to your tables and for providing a place for separate CSS styling.

When printing out a table that doesn't fit onto one (paper) page, most browsers repeat the contents of <thead> on every page.

There's a specific order that must be adhered to, and we should be aware that not every element falls into place as one would expect. The following example demonstrates how our 4 elements should be placed.

<table>

<caption>Table Title</caption> , like so:

<!— I’m an HTML comment! —>

They can be incorporated inline within other content:

<h1>This part will be displayed